06

Viajando en el tiempo

Transcripción

[00:00] Hola todos y todas. Bienvenidos y bienvenidas una vez más a este curso de Git. Como comentamos en el último video, quería dar un análisis al proyecto como un todo en el momento que hice ese commit con un merge en la branch lista. ¿Cómo será que puedo viajar en el tiempo y ver lo que ocurrió en ese momento?

[00:18] Antes que nada, recuerden que hicimos un git revert de un commit utilizando un hash grande, pero podemos ejecutar ese mismo revert utilizando solo los primeros siete caracteres del commit al que vamos a volver, porque normalmente esos siete caracteres son suficientes para identificar unívocamente a un commit.

[00:42] Por ejemplo, vean acá en este git log que simplemente tenemos los siete primeros caracteres, entonces eso nos da un indicio de que con eso es suficiente para identificar cada uno de esos commit. Entonces, quiero navegar por mi proyecto hasta llegar al estado que tenía cuando hice el commit de merge con la rama lista.

[01:01] Entonces voy a copiar el hash, este hash del commit, estos siete primeros caracteres y vamos a hacer un git checkout y vamos a colocar paste. Una vez que hago enter, vean el mensaje que nos está diciendo. No está diciendo que ahora estamos en un modo desanexado del HEAD, de nuestro estado HEAD que es el lugar en donde nosotros estamos parados.

[01:28] Dice que podemos por ejemplo mirar, experimentar, hacer cambios, commitear sin problemas y esos commits, todos esos cambios van a ser descartados cuando por ejemplo volvamos a nuestra branch master. Entonces por ejemplo, si nosotros quisiéramos mantener los cambios que vamos a hacer acá dentro de este estado desanexado del HEAD en este commit en particular.

[01:57] Lo que podemos hacer, nos dice acá, es utilizar una nueva branch, crear una nueva branch, y después de ahí sí hacer un commit en esa nueva branch. Vamos a verlo un poco mejor dentro de nuestro Visualizing Git. Acá estamos dentro de Visualizing Git, vamos a hacer un ejemplo, supongamos que estamos haciendo commits, git commit, git commit, enter y supongamos que nosotros nos queremos volver a este segundo commit de acá.

[02:30] Entonces vamos a hacer un git checkout y vamos a colocar el hash que tenemos, que nos está presentando ahí. 162815e, enter. Vean que hemos vuelto, nuestro HEAD ha vuelto a ese commit anterior, pero no está dentro de ninguna branch, o sea no está acá adentro de master, está colocado encima del commit. Por ejemplo, si yo hiciera un git commit ahora vean lo que ocurre.

[03:01] Se ha creado un nuevo commit en un lugar que no tiene nombre. Este commit no pertenece a ninguna línea de desarrollo. Si por ejemplo nosotros volviéramos al master, al git checkout master, nosotros no podríamos volver nunca a ese commit que tenemos ahí. Entonces lo que nos estaba diciendo dentro de la terminal es que si hacemos, vamos a volver de nuevo al segundo commit, git checkout 162815e, enter.

[03:37] Si los cambios que nosotros vamos a hacer los queremos conservar, entonces tenemos que hacer primero un git checkout -b para crear una nueva branch. Enter. Fíjense que ahora estamos dentro de una nueva branch. Ahora sí por ejemplo si hacemos git commit, damos enter, vean que ese commit ya pertenece a la nueva branch.

[04:06] Y por ejemplo nosotros podemos hacer un git checkout master, fuimos para master, podemos volver al nuevo commit, a la nueva branch, git checkout nuevaBranch, enter, y podemos ir de nuevo a ese commit que nosotros creamos a partir de un commit viejo. Ya puedo navegar entre los estados de mi aplicación.

[04:35] Vamos a volver a nuestra terminal y vamos a hacer un git log --oneline, enter. Y vean dónde estoy. Estoy parado cuando hicimos un merge con la branch lista. Si nosotros vamos ahora al código, vean que no están hechos, no están modificados ni el curso de Vagrant, ni Ansible ni Kubernetes. Es decir, realmente volví atrás en el tiempo.

[05:06] Aún no ocurrieron esas actualizaciones. Si quiero puedo jugar acá con el código, puedo modificar todo lo que quiera, guardar y cuando esté listo para volver a mi último commit dentro de mi branch master, puedo hacer un checkout para master y listo.

[05:24] Vean que ya tengo, puedo hacer así. Vamos a hacer un git checkout master, enter y vean ahora el código, ya tenemos todo el código actualizado con lo último que hemos hecho. De esa forma, podemos navegar por nuestra aplicación y con esto ya tenemos bastante conocimiento. Podemos hacer casi cualquier cosa necesaria para trabajar en un día normal, usando un controlador de versiones.

[05:52] Pero, ¿cómo informo "acá tengo una versión lista en mi sistema"? ¿Cómo le digo a Git eso: "acá tengo un entregable, un release"? ¿Será que Git nos ayuda a generar un entregable, un proyecto listo? Vamos a hablar sobre eso en el próximo capítulo. Pero antes, sáquense todas las dudas que tengan, hagan los ejercicios prestando atención y después de que estén bien seguros, los espero en el próximo capítulo con un contenido bastante interesante.